

Formål

Hovedformålet med denne øvelse er at anvende viden om instruktionernes eksekveringstid til at skrive kode, der giver en ønsket tidsforsinkelse.

Tidsforsinkelserne anvendes i denne øvelse til at spille "musik" ved hjælp af en lyd giver.

Materiale

Lærebogen (Mazidi): Ugens lektie.

Lektion 5: "Assembly programmering".

Lektion 6: "Status register og delays".

Øvelsen

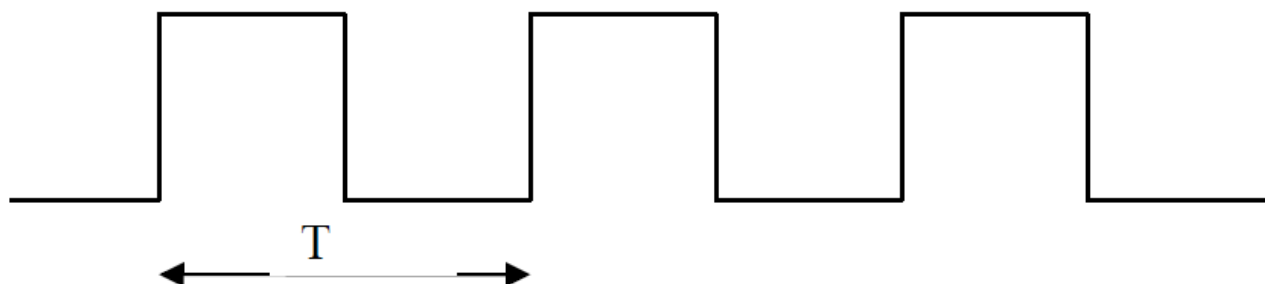
I denne øvelse vil vi:

- Skrive kode, der laver et delay på 4 mikrosekunder.
- Anvende en lyd giver til at spille toner med en bestemt frekvens.
- Skrive et program, der spiller en melodi på C-dur skalaen.

I denne øvelse vil vi lave praktiske forsøg, der underbygger teorien omkring delay sløjfer.

Til brug for øvelsen anvendes den lille højttaler, som er monteret på "Arduino Mega2560" kittet.

Hvis lyd giveren tilføres et firkantsignal med en given frekvens, vil der høres en tone, hvis grundfrekvens er lig med signalets frekvens (frekvensen $f = 1/T$, hvor T er periodetiden).



Figur 1

Vi ønsker at anvende lyd giveren til at spille tonerne, der svarer til en C dur skala:



Figur 2

I tabellen nedenunder ses, hvilke frekvenser der svarer til de 8 toner i C dur skalaen:

Tone	Frekvens	$T = 1/f$	$T/2$	$(T/2)/(4\mu s)$
c	523,25 Hz	1911 μs	956 μs	239
D	587,33 Hz	1792 μs	851 μs	213
E	659,26 Hz	1517 μs	758 μs	190
F	698,46 Hz	1432 μs	716 μs	179
G	783,99 Hz	1276 μs	638 μs	160
A	880,00 Hz	1136 μs	568 μs	142
H	987,77 Hz	1012 μs	506 μs	127
C	1046,50 Hz	956 μs	478 μs	120

Tabel 1: Frekvenserne + beregninger for en C dur skala.

I kolonnen " $T = 1/f$ " er periodetiden udregnet (i mikrosekunder) og i kolonnen " $T/2$ " ses tiderne for en halvperiode (igen i mikrosekunder).

I den sidste kolonne " $(T/2)/(4 \mu s)$ " ses hvor mange tidsforsinkelser på 4 mikrosekunder, der skal til for hver halvperiode af tonen. Årsagen til denne udregning vil fremgå, når vi skal skrive vores program.

Opret et assembly projekt i Atmel Studio (kald det LAB3_1) og skriv følgende kode (eller copy/paste fra filen "LAB3_1.txt", der ligger på Brightspace):

```
;***** INITIERING *****
;***** Klargoer SP og PORTB *****
    LDI R16,HIGH(RAMEND) ;Klargoer Stack Pointer
    OUT SPH,R16
    LDI R16,LOW(RAMEND)
    OUT SPL,R16
    SER R16              ;PORT B er udgang
    OUT DDRB,R16
    CLR R16              ;Sluk alle lysdioder
    OUT PORTB,R16

;***** PROGRAM LOOP *****
    LDI R20,239 ;c      Spil C dur skalaen
    CALL TONE
    LDI R20,213 ;D
    CALL TONE
    LDI R20,190 ;E
    CALL TONE
    LDI R20,179 ;F
    CALL TONE
    LDI R20,160 ;G
    CALL TONE
    LDI R20,142 ;A
    CALL TONE
    LDI R20,127 ;H
    CALL TONE
    LDI R20,120 ;C
    CALL TONE
HERE:
    JMP HERE ;Bliv her "altid"

;*****
;* DELAY (R18*4us) *
;* Laver en tidsforsinkelse paa *
;* "Det der er i R18" gange 4 us *
;*****
DELAY:
    LDI R17,xxx ;<--- OPGAVE (del 1): Find xxx og skriv det her
AGAIN:
    DEC R17
    BRNE AGAIN
    DEC R18
    BRNE DELAY
    RET

;*****
;***** SPILLER EN TONE *****
;***** 250 HALVPERIODER *****
;***** T/2 = (R20)*4 us *****
;*****
TONE:
    LDI R18,200 ;<--- Fjern denne instruktion, inden del 2 laves
                ;<--- HER MANGLER KODE (del 2)
    CALL DELAY ;Delay = (R18) * 4 mikrosekunder
                ;<--- HER MANGLER KODE (del 2)

    RET ;Retur til program loop
```

Programmet skal spille C dur skalaen på lydgiveren, men er dog ikke færdigskrevet (pilene i programkommentarerne).

Brug lidt tid på at overveje, hvordan programmets funktion er tænkt.

Lydgiveren på I/O shieldet anvendes, og husk at placere buzzer jumperen rigtigt:



Figur 3

Når jumperen er placeret ved teksten "ON", er lydgiveren tilkoblet.

Når jumperen er placeret ved teksten "OFF", er lydgiveren frakoblet.

Del 1

Vi ønsker at kalde funktion DELAY (fra funktionen TONE) , når en tone skal afspilles.

DELAY-funktionen er skrevet, men som det ses, er der endnu ikke taget stilling til, hvilken værdi, R17 skal loades med. Find denne værdi og modifier instruktionen ”LDI R17,xxx”.

Beregn værdien ”xxx” med bedst mulig nøjagtighed, sådan at funktionen giver et delay på cirka:

Delay = (”tallet i R18”) * 4 mikrosekunder.

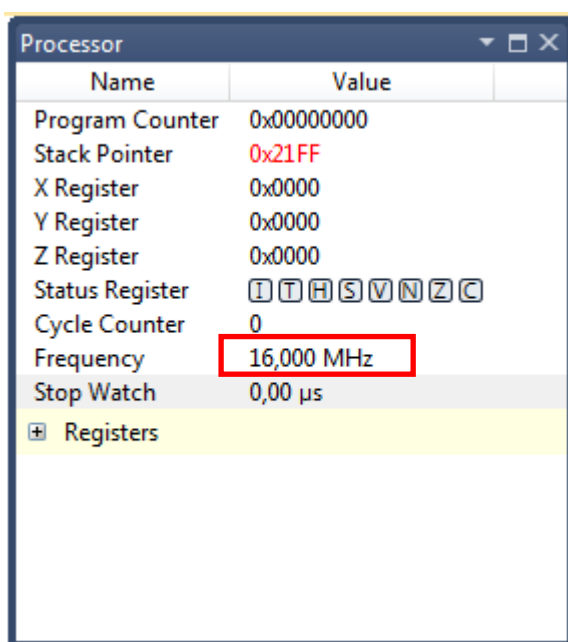
Anvend din viden om de enkelte instruktioners eksekveringstid.

Test herefter, med brug af Microchip Studios simulator, at din beregning passer.

Anvend ”Stop Watch” i simulatorens ”Processor”-vindue til at udmåle tiden.

Stopuret kan nulstilles, når man vil - blot ved at højreklikke på ”Stop Watch” og vælge ”Reset Stopwatch”.

OBS: For at stopuret måler rigtigt, er det vigtigt, at man har indstillet den rigtige clock-frekvens (altså den som vores ”Arduino Mega2560” anvender). **Denne frekvens er 16 MHz.**



Figur 4

Frekvensen indtastes simpelt ved at klikke i feltet ”Frequency” og derefter editere tallet.

Del 2

Skriv den manglende kode for funktionen TONE.

Husk at starte med at slette instruktionen "LDI R18,200", som blot var en hjælp til at få løst del 1.

Som det ses i programmets hovedsløjfe loader vi R20 med en værdi, hvorefter vi kalder TONE for at få afspillet en tone:

```
;***** PROGRAM-LOOP *****  
LDI R20,239 ;c  
CALL TONE  
LDI R20,213 ;D  
CALL TONE
```

Værdien, som vi loader til R20, er netop værdien fra tabel 1 (altså tones halvperiode delt med 4 mikrosekunder). Med andre ord: Hvor mange gange 4 us delay, der skal være per halvperiode for at få den tonefrekvens, som vi ønsker.

Funktionen TONE skal afspille den ønskede tone ved at "toggle" (= invertere) PORTB, vente en tid svarende til en halv toneperiode, og så toggle PORTB igen.
I alt skal der afspilles 250 halvperioder (det giver en "passende" længde for tonen).

Koden for funktionen TONE kan skrives efter følgende trinvis opskrift:

1. Load et register (som vi endnu ikke har brugt i programmet) med 250.
2. Complement et register (som vi endnu ikke har anvendt i programmet), og anvend en OUT instruktion til at sende registerets værdi ud på PORTB.
3. Brug en MOV instruktion til at kopiere R20 til R18.
4. Brug en CALL instruktion til at kalde funktionen DELAY (er allerede skrevet).
5. Dekrementer registeret nævnt under punkt 1.
6. Brug en BRNE instruktion til at hoppe til punkt 2, hvis dekrementeringen ikke gav 0.
7. Returner fra funktionen (via RET instruktionen, som allerede er skrevet).

Afprøv herefter det færdige program ved at downloade det til "Arduino Mega2560" (skal spille en C dur skala, hver gang du RESET'er din Mega2560).

Del 3 (ekstraopgave)

Lav ændringer i programmets hovedsløjfe, sådan at der spilles en melodi i stedet for blot C dur skalaen. "Lille Peter edderkop" skulle være kendt af de fleste ☺. "Smoke on the Water" (eller noget lidt i den retning) er også hørt ☺.

Del 4 (ekstraekstraopgave)

Programmet er relativt simpelt opbygges, idet en tones længde er defineret til at være 250 halvperioder. Tonelængden kommer derfor til at være afhængig af tonens frekvens, hvilket jo ikke er optimalt.

Overvej, om du kan finde en metode til at undgå denne ulempe.